VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Alžbeta Kovaľová

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV RADIOELEKTRONIKY
DEPARTMENT OF RADIO ELECTRONICS

INDIKÁTOR STAVU ROBOTICKÉHO SYSTÉMU
THE ROBOTIC SYSTEM STATE INDICATOR

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Alžbeta Kovaľová**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Roman Šotner, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Elektronika a sdělovací technika**
Ústav radioelektroniky

Studentka: Alžbeta Kovaľová ***ID:*** 195362
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Indikátor stavu robotického systému

POKYNY PRO VYPRACOVÁNÍ:

Cílem této práce je návrh a realizace zařízení pro indikaci stavu robotického systému komunikujícího pomocí rozhraní CAN a využívající protokol vyvinutý ve firmě Y Soft. Seznamte se s vlastnostmi a možnostmi systému pro robotické testování embedded zařízení a definujte požadavky na výsledné zařízení. Vyberte vhodný mikrokontroler a seznamte se se sběrnicí CAN a protokolem využívaným ve firmě Y Soft. Navrhněte a sestavte koncept prototypu zařízení demonstrujícího možnosti indikace stavu pomocí různých periférií.

Navrhněte způsob komunikace s nadřazeným systémem pro testování zařízení. Připravte plošné spoje, naprogramujte mikrokontroler a zprovozněte komunikaci pro sběrnici CAN. Integrujte vytvořené zařízení do firemního systému komplexního robotického testování, jeho funkčnost ověřte a vyhodnoťte.

DOPORUČENÁ LITERATURA:

[1] BARR, Michael, MASSA, Anthony. Programming embedded systems: with C and GNU development tools. 2nd ed. Sebastopol: O'Reilly, 2006. ISBN 9780596009830

[2] YIU, Joseph. ARM® Cortex®-M for Beginners: An overview of the ARM Cortex-M processor family and comparison. ARM Community [online]. Cambridge, UK: ARM, 2016, 21.6.2017 [cit. 2018-07-20]. Dostupné z: https://community.arm.com/processors/b/blog/posts/white-paper-cortex-m-for-beginners-an-overview--f-the-arm-cortex-m-processor-family-and-comparison

Termín zadání: 4.2.2019 ***Termín odevzdání:*** 23.5.2019

Vedoucí práce: doc. Ing. Roman Šotner, Ph.D.
Konzultant:

prof. Ing. Tomáš Kratochvíl, Ph.D.
předseda oborové rady

Abstrakt

Táto bakalárska práca sa zaoberá návrhom a zhotovením indikátora stavu robotického systému využívajúceho vo firme Y Soft. Súčasťou tejto práce je návrh a realizácia komunikácie medzi indikátorom stavu s nadradeným systémom pre testovanie zariadení. Je navrhnutý a skonštruovaný plošný spoj a následne je naprogramovaný mikrokontrolér a zrealizovaná komunikácia po zbernici CAN. Následne je vytvorené zariadenie integrované do firemného systému pre robotické testovanie, kde je overená jeho funkčnosť.

Klíčová slova

Robotický systém, mikrokontrolér, komunikačný protokol, SPI, zbernica CAN, vstavaný systém, indikátor.

Abstract

This bachelor thesis deals with the design and creation of a status indicator for robotic system developed in Y Soft. Part of this work is the design and implementation of communication between the status indicator and the superior device testing system. A printed circuit board is designed and constructed, and then a microcontroller is programmed and the CAN bus communication is implemented. Subsequently, the device is integrated into the company system for robotic testing, where its functionality is verified.

Keywords

Robotic system, microcontroller, communication protocol, SPI, CAN bus, embedded system, indicator.

Bibliografická citace:

KOVAĽOVÁ, Alžbeta. Indikátor stavu robotického systému [online]. Brno, 2019 *[cit. 2019-05-13]*. Dostupné z: *https://www.vutbr.cz/studenti/zav-prace/detail/118414*. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky. Vedoucí práce Roman Šotner.

Prohlášení

„Prohlašuji, že svou bakalářskou práci na téma Indikátor stavu robotického systému jsem vypracovala samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědoma následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: …… ……
podpis autora

Poděkování

Týmto sa chcem poďakovať vedúcemu mojej bakalárskej práce vo firme Y Soft Ing. Václavovi Novotnému, môjmu vedúcemu bakalárskej práce v škole doc. Ing. Romanovi Šotnerovi, Ph.D. a v neposlednom rade kolegovi Bc. Otovi Dušekovi za odborné rady, trpezlivosť a povzbudivé slová pri spracovaní mojej bakalárskej práce.

V Brně dne: …… ……
podpis autora

Obsah

Zoznam symbolov a skratiek

Skratky:

ADC … Analog To Digital Converter

ALU … Arithmetic Logic Unit; aritmeticko logická jednotka

AMP … Arbitration on Message Priority; arbitráž na prioritu správy

API … Application Programming Interface; aplikačné programovacie rozhranie

CAD … Computer Aided Design; počítačom podporovaný dizajn

CAN … Controller Area Network

CD … Collision Detection; detekcia kolízií

CISC … Complex Instruction Set Computing; počítač s rozsiahlou inštrukčnou súpravou

CMOS … Complementary Metal Oxide Semiconductor; technológia používaná pri výrobe čipov a mikroprocesorov

CRC … Cyclic Redundancy Check; kontrola cyklickým kódom

CSMA … Carrier Sense Multiple Access; viacnásobný prístup nosičov

DAC … Digital to Analog Converter

DPS … Doska Plošných Spojov

DSP … Digital Signal Processor; digitálny signálový procesor

ECU … Electronic Control Unit; elektronická riadiaca jednotka

EOF … End Of Frame; koniec rámca

ESD … Electrostatic Discharge; elektrostatický výboj

FPU … Floating Point Unit; jednotka s pohyblivou rádovou čiarkou

GPIO … General purpose input/output; všeobecne použiteľný vstup/výstup

HAL … Hardware Abstraction Layer

IDE … Integrated Development Environment; integrované vývojové prostredie

IFS … InterFrame Space; medzirámový priestor

IoT … Internet of Things

I2C … Inter integrated Circuit

LSB … Least Significant Bit; najmenej významný bit

MCU … MicroController Unit

MISO … Master Input / Slave Output

MOSI … Master Output / Slave Input

MPU … Memory Protection Unit; jednotka ochrany pamäte

MSB … Most Significant Bit; najviac významný bit

NVIC … Nested Vectored Interrupt Controller; vstavaný riadiaci vektorový prerušovač

QA … Quality Assurance; zabezpečenie kvality

PCB … Printed Circuit Board; doska plošných spojov

PWM … Pulse-Width Modulation; impulzová šírková modulácia

REST … REpresentional State Transfer; architektúra rozhrania

RGB … Red Green Blue

RISC … Reduced Instruction Set Computer; počítač s redukovanou inštrukčnou súpravou

RTC … Real Time Clock

RTR … Remote Request; vzdialená požiadavka

SCLK … Serial CLocK; sériové hodiny

SDA … Serial Data; sériové dáta

SOC … System On a Chip; systém na jednom čipe

SOF … Start Of Frame; začiatok rámca

SPI … Serial Peripheral Interface; sériové periférne rozhranie

SRR … Substitute Remote; náhradné diaľkové ovládanie

SS/CS … Slave Select / Chip Select

SWD … Serial Wire Debug; sériové ladenie kábla

UART … Universal Asynchronous Receiver Transmitter; univerzálny asynchrónny prijímač/vysielač

WIC … Wakeup Interrupt Controller; ovládač prerušenia budenia

Zoznam obrázkov

ÚVOD

Cieľom tejto práce je navrhnúť a zrealizovať zariadenie pre indikáciu stavu robotického systému. Robotický systém je súčasťou jedného z projektov firmy Y Soft, ktorá sa zaoberá najmä správou tlače, ku ktorej patrí aj 3D tlač a digitalizáciou dokumentov. Jeden z hlavných komponentov využívajúcich robotický systém je aplikácia YSoft SafeQ, ktorá sa testuje na multifunkčných tlačiarňach, 3D tlačiarňach, mobilných zariadeniach s dotykovým displejom a IoT (Internet of Things) platformách.

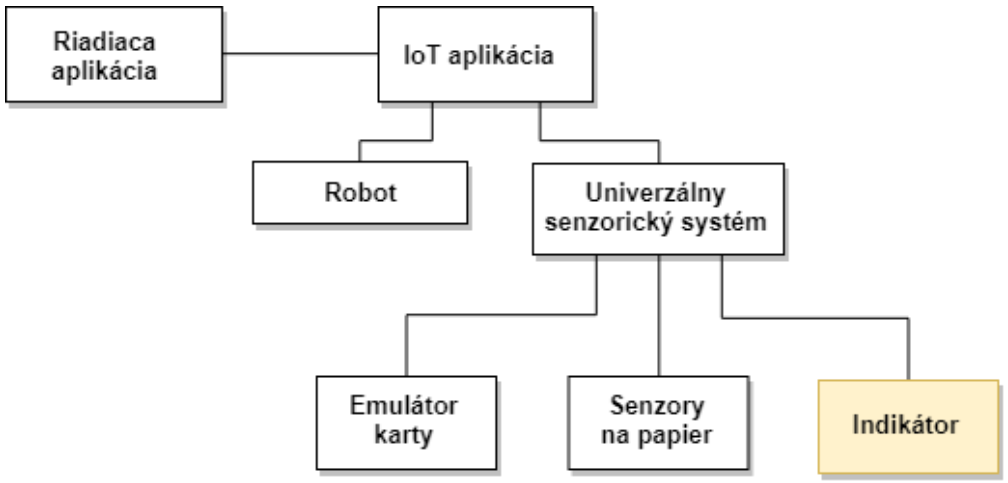
YSoft SafeQ aplikácia vydáva nové aktualizácie, pričom každá nová aktualizácia vyžaduje dôkladné a v ideálnom prípade čo najrýchlejšie otestovanie. Testovanie jednotlivých zariadení podporujúcich túto aplikáciu trvá QA (Quality Assurance) inžinierom približne 2 týždne, čo sa za pomoci robotického systému skráti na 2 dni. Je to možné vďaka neustálej prevádzke robota, ktorý pracuje 24 hodín denne, 7 dní v týždni.

Robotický systém zefektívňuje dobu testovania prostredníctvom neustálej prevádzky robota, ktorá je možná vďaka automatizovanej inštalácii a paralelnému testovaniu podporovaných zariadení. Neprestajná prevádzka robota prináša výhodu aj z hľadiska rýchlej spätnej väzby. Akákoľvek zmena aplikácie je hneď otestovaná a zaznamenaná počas nočného testovania a prináša inžinierom informácie o behu hneď na druhý deň.

Celý robotický systém pozostáva z niekoľkých hardvérových a softvérových komponentov, ku ktorým patria: robot, senzory na papier, emulátor karty, kamera, IoT platformy, aplikácie pre získavanie a spracovanie obrazu a ďalšie prvky sprostredkujúce komunikáciu medzi jednotlivými komponentami. Vývoj celého systému napreduje a s tým tiež narastá počet testovaných zariadení. Avšak pri testovaní je potrebné sledovať aj stav robotického systému, ktorý sa kedykoľvek môže dostať do chybného stavu. Tento chybný stav môže spôsobiť zmeravenie systému, ktoré nemusí byť na prvý pohľad zjavné, hlavne pri veľkom počte testovaných zariadení. Pomocou indikátora stavu je na prvý pohľad viditeľné, že došlo k chybe v systéme, čo je späté s jednoduchším monitorovaním celého robotického systému a následnou diagnostikou.

1. ROZBOR ZADANIA

Cieľom bakalárskej práce je návrh komunikácie indikátora stavu robotického systému pomocou zbernice CAN s nadradeným systémom, navrhnúť a skonštruovať dosku plošných spojov, naprogramovať mikrokontrolér a oboznámiť sa s komunikačným protokolom používaným vo firme Y Soft. Po skonštruovaní a sprevádzkovaní komunikácie nasleduje samotná integrácia vytvoreného zariadenia do firemného systému a overenie funkčnosti pri robotickom testovaní.



Obr. 1.1-1 Schéma robotického systému

1.1 Postup návrhu

Navrhnutý indikátor má zahŕňať funkcie pre rozoznanie viacerých druhov stavov, v ktorých sa môže ocitnúť robotický systém a zároveň musí spoľahlivo komunikovať so zvyšnými perifériami celého systému, aby bola indikácia hodnoverná. Preto návrh začne vhodným výberom mikrokontroléra a signalizačných periférií. Následne dôjde k samotnému vytvoreniu DPS (Doska Plošných Spojov), naprogramovaniu mikrokontroléra a komunikácií s jednotlivými komponentami a nadradeným systémom.

1.2 Požiadavky

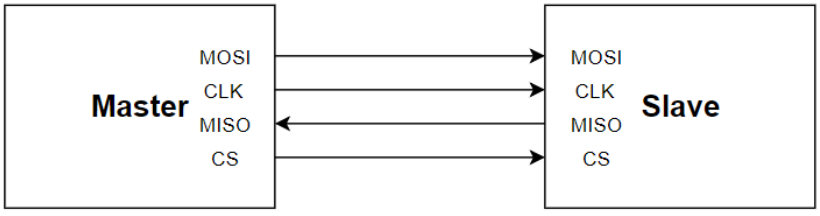
Navrhované zariadenie je súčasťou komplexnejšieho systému, preto hlavná požiadavka je zameraná na kompatibilitu so zvyšným systémom. Výber komunikačného protokolu a rozhrania sú podmienené už existujúcou komunikáciou zabudovanou v robotickom systéme. Dôležitá je stabilita a jednoduchosť konštrukcie pre flexibilnejšiu prácu so zariadením. A v neposlednom prípade je dobré myslieť aj na vonkajšiu štruktúru zariadenia, aby bola detekcia stavu zjavná v akýchkoľvek podmienkach (počas dňa alebo noci).

2. TEORETICKÝ ÚVOD

V tejto kapitole sú uvedené základné teoretické informácie k problematike návrhu a realizácie zariadenia pre indikáciu stavu robotického systému. Najprv je potrebné zmieniť sa o princípe komunikácie medzi jednotlivými periférami sériového rozhrania a neskôr sa zoznámiť s komunikačným protokolom zbernice CAN (Controller Area Network).

2.1 Sériové periférne rozhranie SPI

SPI (Serial Peripheral Interface) je komunikačný protokol využívajúci pre komunikáciu medzi riadiacimi mikroprocesormi a ostatnými integrovanými obvodmi. Výhodou je skutočnosť, že dáta môžu byť prenášané bez prerušenia. Akýkoľvek počet bitov môže byť odoslaný alebo prijímaný v neprestajnom toku na rozdiel od protokolov I2C (Inter Integrated Circuit) a UART (Universal Asynchronous Receiver Transmitter), kde sa dáta posielajú v paketoch, ktoré sú obmedzené na konkrétny počet bitov. Mnoho mikrokontrolérov má vbudované SPI, ktoré detailne a rýchlo riadi posielanie a prijímanie dát. SPI tiež poskytuje plne duplexnú synchrónnu komunikáciu medzi hlavným zariadením (master) a vedľajším (slave) použitím dvoch dátových liniek. Master vo väčšine prípadov poukazuje na mikrokontrolér, kým slave zvyčajne na senzory, displeje či ADC (Analog to Digital Converter). Vzhľadom k tomu, že SPI nie je štandardizované, môžu nastať situácie, v ktorých sa prenášajú informácie od MSB (Most Significant Bit) po LSB (Least Significant Bit) alebo opačne od LSB po MSB. Najjednoduchšia konfigurácia SPI pozostáva zo Single Master, Single Slave systému, ktorého ilustrácia je znázornená na obrázku Obr. 2.1-1. Medzi základné 4 linky patria: MOSI (Master Output/Slave Input) – linka pre hlavné zariadenie odosielajúce dáta do vedľajších, MISO (Master Input/Slave Output) – linka pre vedľajšie zariadenie posielajúce dáta do hlavného, SCLK (Serial Clock) – hodinový signál a SS/CS (Slave Select/Chip Select) – linka pre hlavné zariadenie, ktoré si vyberá konkrétne vedľajšie zariadenie, na ktoré odošle dáta. Tieto informácie s ešte bližším vysvetlením sú spomenuté v literatúre [1].



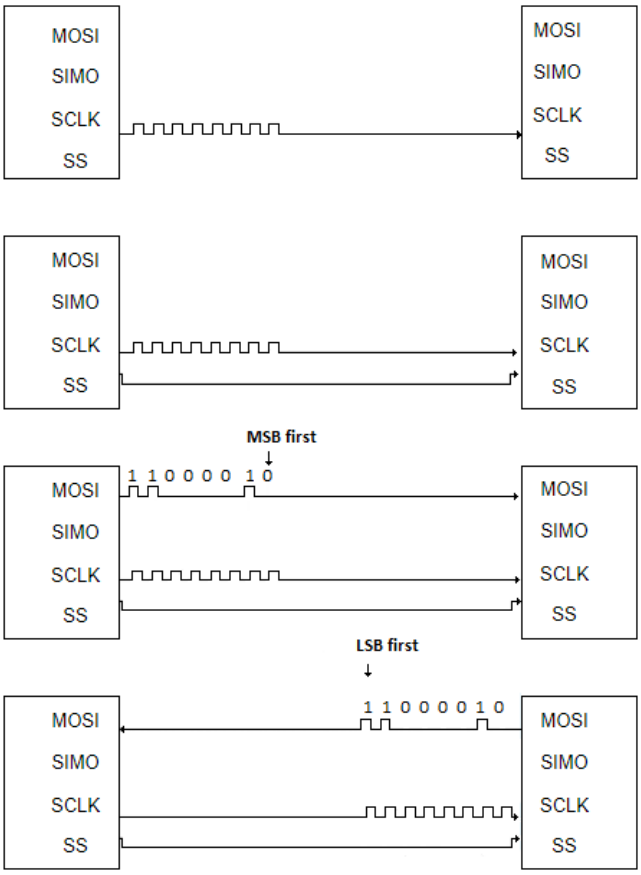
Obr. 2.1-1 Najjednoduchšia konfigurácia SPI [1]

2.1.1 Princíp funkčnosti SPI

Literatúra [1] a [2] tvoria informačný prameň pre vysvetlenie tejto problematiky. Hodinový signál, ktorý vždy iniciuje master, riadi výstup dátových bitov z mastra s odberom vzoriek bitov na slave. Jeden bit sa prenáša v každom hodinovom cykle, z čoho vyplýva, že rýchlosť prenášania dát je určená frekvenciou hodinového signálu. Hodinový signál môže byť menený pomocou vlastností polarity a fázy hodinového signálu. Polarita môže byť v neivertujúcom alebo invertujúcom stave. Neinvertujúci stav určuje pozíciu hodinového signálu v nízkej úrovni pri poklese SS/CS do nízkej úrovne počas zahájenia komunikácie. Ivertujúci stav zase nastaví hodinový signál do vysokej úrovne pri zahájení komunikácie. Fáza signálu je nakonfigurovaná tak, že dáta sú buď posúvané alebo vzorkované na nástupnej alebo vzostupnej hrane hodinového signálu. Slave select poukazuje na to, že master si môže zvoliť, s ktorým slave chce komunikovať, nastavením linky SS/CS na nízku úroveň napätia pre konkrétneho slave. V neprenosnom stave je slave udržiavaný vo vysokej úrovni napätia. SPI môže byť nastavené tak, aby fungovalo s jedným hlavným zariadením a jedným vedľajším alebo aj s viacerými vedľajšími. V prípade MOSI slave prijíma dáta odoslané z mastra na pine $MOSI$. Dáta poslané z mastra na slave sa zvyčajne posielajú najprv s MSB. Slave môže opačne odoslať dáta mastrovi cez pin $MISO$. V tomto prípade sa naopak zvyčajne ako prvé posielajú s LSB. Veľkou výhodou oddelených MISO a MOSI je, že umožňujú súčasné odosielanie a prijímanie dát.

2.1.2 Postup prenosu dát

Postup prenosu dát znázornený na obrázku Obr. 2.1-2 začína tým, že master vyšle hodinový signál z pinu $SCLK$ na slave, potom nastaví na SS/CS pine nízku úroveň napätia pre stav, ktorý aktivuje slave a pripraví dáta k prenosu do posuvných registrov. Master pošle dáta do slave cez linku $MOSI$ a ak je potrebná odpoveď, tak slave vracia dáta do mastra cez linku $MISO$. Po odoslaní dát dôjde k zastaveniu hodinového signálu a nastane možnosť generovania prerušenia. Ak sa odošlú všetky dáta, slave sa odpojí, čo znamená, že pin SS/CS prejde do vysokej úrovne. Viac informácií ohľadom prenosu dát je zmienených v zdrojoch [1] a [3].



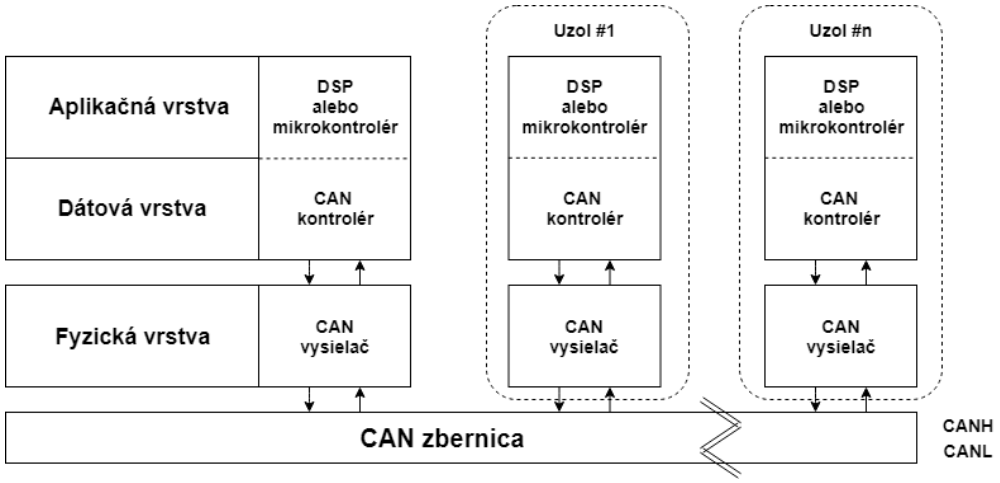
Obr. 2.1-2 Postup prenosu dát pre SPI [1]

2.2 Zbernica CAN

Zbernica CAN predstavuje dvojvodičovú sériovú dátovú zbernicu lokálnej siete, ktorá umožňuje mikrokontrolérom a ďalším funkčným jednotkám komunikovať medzi sebou (Obr. 2.2-1). Patrí medzi robustné systémy a je odolná voči poruchám subsystému a elektromagnetickému rušeniu vďaka využitiu dvoch diferenciálnych vodičov. Najviac sa využíva v automobilovej technike, kde v súčasnej dobe predstavuje štandard prakticky pre všetky vozidlá, počínajúc osobnými autami, končiac radarovými systémami či ponorkami. Využíva sa taktiež v oblasti automatizácie alebo medicíny. Táto zbernica je tiež známa tým, že pre komunikáciu nemusí využívať mastra. Ide o tzv. multi-mastera, ktorý by riadil komunikáciu na zbernici. Takže jednotlivé zariadenia pripojené na túto zbernicu môžu komunikovať medzi sebou a taktiež môžu súčasne začať vysielať dáta bez toho, aby došlo k nejakej kolízii. Bližší prehľad informácií o CAN zbernici je spomenutý v literatúre [4].

Princíp funkčnosti zbernice je založený na pridelení unikátneho identifikátora ku každej správe, ktorého súčasťou je aj informácia o priorite, podľa ktorej sa rozhodne o význame správy. Každá ECU (Electronic Control Unit) obsahuje čip pre príjem všetkých odoslaných správ a rozhoduje o ich relevantnosti. Rieši

aj konflikty, ktoré môžu nastať pri súčasnom vysielaní správ z viacerých uzlov. Štandard CAN zaisťuje kontrolu obsahu správy pomocou kódu na detekciu chýb CRC (Cyclic Redundancy Check) [5]. V prípade, že dôjde k poškodeniu správy, zariadenia odošlú chybovú správu, aby došlo k opätovnému odoslaniu. Na obrázku Obr. 2.2-1 je vidieť základné komponenty CAN zbernice, ktoré sa členia do troch vrstiev. Najnižšia fyzická vrstva je tvorená samotným vysielačom. Nasledujúcu vrstvu tvorí kontrolér, ktorý sa stará napr. o detekciu chýb alebo potvrdenie stavu správy. Najvyššia vrstva, aplikačná, pozostáva z už konkrétnej aplikácie. Tieto a podrobnejšie informácie ku CAN sa nachádzajú v literatúre [6].



Obr. 2.2-1 Bloková schéma CAN zbernice [6]

2.2.1 Arbitráž

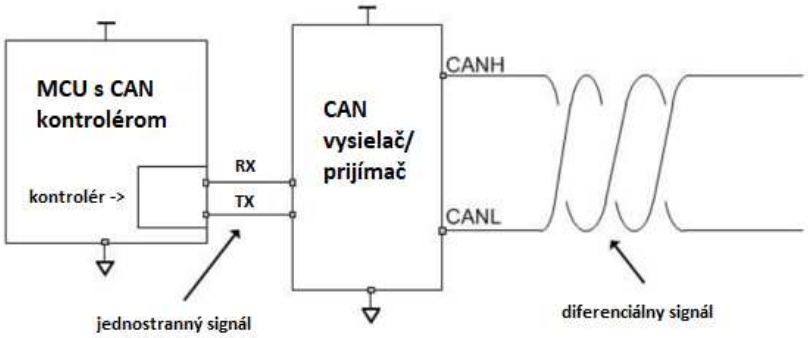
Keďže zbernica CAN je protokol s viacerými prístupmi, je nutné, aby sledovala vysielanie dát. V prípade detekcie dát na zbernici zastaví vysielanie vlastných dát a čaká istú dobu, po ktorej zopakuje kontrolu. Tento chod zakazuje narušenie prebiehajúcej komunikácie. Avšak môže nastať situácia, kedy začnú dva uzly vysielať dáta v rovnakom čase. V takomto prípade uvedený mechanizmus kontroly nebude užitočný. Na vyriešenie tohto problému používa CAN detekciu kolíziou a bitovú arbitráž [4].

Pre porozumenie arbitráže je potrebné bližšie špecifikovať prenášanie dát na zbernici CAN, čo je zdokumentované v [4]. Dáta sa prenášajú cez uzly vo forme rámcov. Rámec má viacero polí, z ktorých podstatné pre prenos sú SOF (Start Of Frame) a identifikátor. SOF identifikuje začiatok rámca a identifikátor typ správy. Čím nižší je identifikátor, tým vyššia je priorita správy. CAN môže byť v recesívnom stave (logická 1) alebo dominantnom stave (logická 0). Zbernica je defaultne v recesívnom stave, takže keď chce vysielať dáta, tak opustí predvolený stav a preklopí sa do dominantného stavu, z ktorého začne vysielať.

2.2.2 Úrovne signalizácie CAN

Ako bolo spomenuté, CAN je sériová zbernica tvoriaca dvoma diferenciálnymi vodičmi. To znamená, že dáta sa vždy posielajú ako jeden bit prostredníctvom dvoch komplementárnych signálov značiacich sa ako CANH a CANL. Vďaka rozdielom potenciálov medzi týmito vodičmi je možné určiť, či je bit dominantný alebo recesívny. Ak je na vodičoch nulový rozdiel potenciálov, tak ide o recesívnu úroveň a v prípade, ak je rozdiel potenciálov nenulový, tak sa nachádza na zbernici dominantná úroveň [7].

Každá CAN aplikácia sa skladá z mikrokontroléra so zabudovaným CAN kontrolérom a vysielačom/prijímačom, ktorý je spojený so zbernicou. Podstatnú úlohu zohráva vysielač/prijímač, tzv. transceiver, ktorým sú spracované dva typy signálov. Ide o jednostranné signály TX a RX a diferenciálne signály CANH a CANL. Takže úlohou tejto časti je pri komunikácii meniť jednostranný signál na diferenciálny a naopak [7].



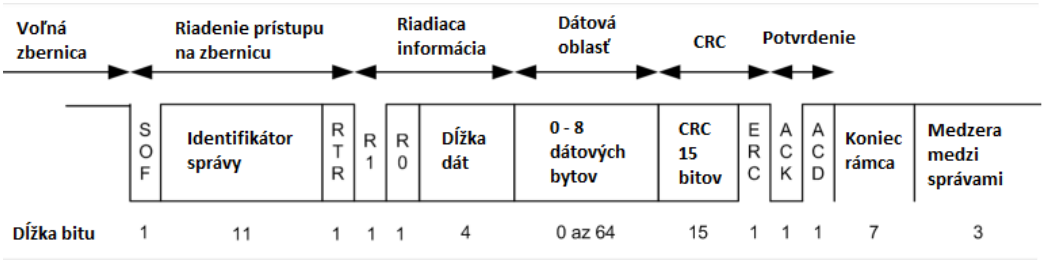
Obr. 2.2-2 CAN kontrolér a CAN vysielač/prijímač [7]

2.2.3 Komunikačný protokol CAN

Komunikačný protokol CAN je protokol s viacerými prístupmi a detekciou kolízií a arbitráže. Ide o CSMA / CD (Carrier Sense Multiple Access / Collision Detection) + AMP (Arbitration on Message Priority), čo znamená , že každý uzol v zbernici musí pred odoslaním čakať na overenie neprítomnosti ďalšej prevádzky, pričom ide o chod zabezpečený opatrením detekciou kolízií a využitia arbitráže [6].

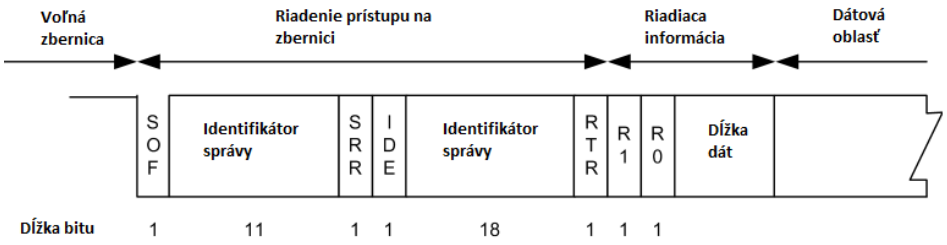
Pri návrhu prototypu a neskôr celého komunikačného systému zabezpečujúceho správny chod pri indikácii stavu robotického systému je nutné zabezpečiť správnu komunikáciu medzi jednotlivými komponentami, ktorú zabezpečuje Y Soft protokol [8]. Základom komunikácie je komunikačný protokol zbernice CAN, ktorá zabezpečuje komunikáciu medzi jednotlivými jednotkami tak, aby nedochádzalo k veľkému zaťaženiu centrálneho procesora. V tomto prípade je využitý rozšírený (extended) dátový formát, ktorého začiatočná časť správy je zobrazená na obrázku

Obr. 2.2-4. Rozdiel oproti štandardnému formátu, ktorý je zobrazený na obrázku Obr. 2.2-3, je v dĺžke identifikátora správy, ktorého 11 bitové pole je rozšírené o 18 bitové pole identifikátora [9].



Obr. 2.2-3 Štandardný formát CAN [9]

Začiatok rámca SOF a identifikátor riadia prístup k zbernici a určujú prioritu správy. RTR (Remote Request) zabezpečuje rozlíšenie stavu správy, ktorá signalizuje chybovosť. Táto signalizácia môže byť reprezentovaná troma spôsobmi. Dominantný (aktívny) stav sa aktívne podieľa na komunikácii po zbernici a tak systém okamžite odhaľuje ľubovoľnú chybu v prenášanej správe. Odošle sa aktívny príznak chyby, ktorý poškodí prenášanú správu. Recesívny (pasívny) stav sa tiež podieľa na komunikácii po zbernici, ale vysiela len informáciu o chybe, ktorá oproti dominantnému stavu nemá deštruktívne účinky na vysielané dáta. V poslednom prípade sa signalizácia správy môže nachádzať v odpojenom stave, ktorý nastane pri tvorbe veľkého množstva chýb. Riadiace pole tvoria dva rezervované bity R0 a R1 a dátové pole, po ktorých nasleduje dátová oblasť tvorená ôsmimi bytmi. Ďalšie polia tvoria kontrolu pre detekciu bitových chýb a po nich nasleduje koniec rámca EOF (End Of Frame). Pokiaľ je RTR bit dominantný, tak na konci rámca sa nachádza pole IFS (InterFrame Space), ktoré slúži pre naviazanie odpovede pri spätnej väzbe. Tento popis jednotlivých polí a informácie zahrnuté vyššie sú čerpané z literatúry [6] a [9].



Obr. 2.2-4 Začiatok dátovej správy rozšíreného formátu CAN [9]

Rozšírený rámec používa 29 bitov pre identifikátor. Týchto 29 bitov je rozdelených do dvoch častí. Prvá obsahuje 11 bitov a druhá 18 bitov. V tomto prípade pole RTR nachádzajúce sa za 11 bitovým poľom identifikátora správy je nahradené poľom SRR (Substitute Remote), ktoré je vždy recesívne a zabezpečuje kompatibilitu so

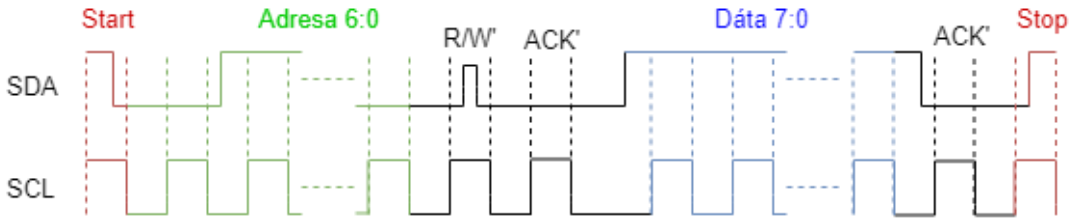
štandardným rámcom. K nemu je pridaný recesívny bit IDE (Identifier Extension), ktorý indikuje nasledujúce rozšírené 18-bitové pole [9].

2.3 Sériová zbernica I2C

Zbernica I2C je dvojvodičová obojstranná zbernica založená na komunikácii, ktorú sprostredkúvajú master a slave [10]. V súčasnej dobe patrí I2C medzi bežne používané zbernice, ktoré nachádzajú uplatnenie v rôznych aplikačných zariadeniach. Najčastejšie sa využíva pri komunikácii so senzormi alebo displejmi. Štrukturálne zahŕňa takmer každý systém inteligentné ovládanie, obvykle jednočipový mikrokontrolér, obvody na všeobecné účely (ovládače LCD, LED, RAM, I/O porty...) alebo obvody zamerané na aplikácie, ktoré sa zaoberajú ladením a spracovaním signálov rádiových a video systémov.
Ako bolo spomenuté, I2C vyžaduje dva vodiče (Obr. 2.3-1), konkrétne vodiče s funkciami dátovej linky SDA (Serial Data) a sériovej hodinovej linky SCL (Serial Clock) [10]. Každé zariadenie pripojené k zbernici je softvérovo adresovateľné jedinečnou adresou a zároveň je uchovaný nekomplikovaný vzťah medzi hlavným a vedľajším zariadením. V prípade prítomnosti viacerých mastrov zúčastňujúcich sa na prenose dát, je zabezpečená detekcia kolízie a arbitráže [10].

2.3.1 Komunikačný protokol I2C

Popis komunikačného protokolu I2C je obsiahnutý v prednáške [3]. SDA a SCL prenášajú informácie medzi zariadeniami pripojenými k zbernici. Každé zariadenie je rozpoznané unikátnou adresou a môže fungovať buď ako vysielač alebo ako prijímač. Komunikáciu na zbernici vždy zahajuje a ukončuje master, ktorého funkciou je zároveň aj generovanie hodinového signálu a vysielanie všetkých požiadaviek. Slave je riadený hodinovým signálom a vždy je adresovaný mastrom. Zahájenie je realizované štartovacím bitom (Start), ktorého hodnota je meniteľná len v okamihu konštantnej vysokej úrovne hodinového signálu. To isté platí v prípade ukončenia prenosu dát, ktoré je zakončené ukončovacím bitom (Stop). Ďalšou podmienkou je, že úroveň dátovej linky musí byť konštantná v okamihu vysokej úrovne hodinového signálu. Celkové rozloženie signálov počas komunikácie je zobrazené na obrázku Obr. 2.3-1. Formát adresného paketu sa potom skladá z deviatich bitov, pričom sedem bitov tvorí adresu podriadeného zariadenia, za ktorou nasleduje riadiaci $R/\overline{W}$ (Read/Write) bit, ktorým sa ovláda zápis alebo vyčítanie dát a potvrdzovací bit $\overline{ACK}$ (Acknowledge), ktorý zabezpečuje oboznámenie o príjme a rozpoznaní adresy. Ak je potvrdzovací bit v nízkej úrovni, tak adresa bola rozpoznaná. Ak sa nachádza vo vysokej úrovni, tak slave je zaneprázdnený a master ukončí komunikáciu.



Obr. 2.3-1 Adresný a dátový paket I2C zbernice [3]

2.4 Mikrokontroléry

Mikrokontrolér alebo MCU je jednočipový počítač tvorený integrovaným obvodom. Často zvyknú byť zakomponované do vstavaných (embedded) systémov, teda vstavaných zariadení a spotrebičov v domácnosti. V dnešnej dobe je široká ponuka rôznych verzií mikrokontrolérov. Výber najvhodnejšieho je závislý od viacerých faktorov. Mikrokontrolér tvorí najpodstatnejšiu časť celého návrhu indikátora, takže je potrebné vziať do úvahy napr. kvalitu vývojových prostriedkov. Z hľadiska schopností periférií, t. j. schopnosti úzko spolupracovať s hardwarom, je STM32 vhodnou voľbou. Ďalšími faktormi je zložitosť programovania alebo dostupnosť a variácia softwarovej knižnice. Okrem iného STM32 obsahuje jadro Cortex-M, o ktorom sú bližšie informácie spomenuté v kapitole 2.4.3. Obsahuje rôzne komunikačné rozhrania, z ktorých podstatné je SPI využité vo výslednom zariadení pri komunikácii so signalizačným komponentom a tiež podporuje CAN štandardnú a rozšírenú verziu protokolu. STM32 zaručujú čiastočnú pinovú kompatibilitu, čo poskytuje jednoduchší prechod na iný mikrokotrolér s minimom aplikačných úprav.

2.4.1 ARM procesor

ARM procesor je jeden z rodiny procesorov založených na architektúre RISC (Reduced Instruction Set Computer) vyvinutá spoločnosťou Advanced RISC Machines (ARM) v 80. rokoch 20. storočia. Nasledujúci rozbor bude vychádzať z [11]. Mikroprocesor označený ako RISC je navrhnutý tak, aby vykonával menší počet typov počítačových inštrukcií a mohol pracovať s väčšou rýchlosťou (viac ako milión inštrukcií za sekundu). Keďže každý typ inštrukcie, ktorý počítač musí vykonať, vyžaduje ďalšie tranzistory a obvody, väčší súbor inštrukcií má tendenciu spomaliť mikroprocesor. Odstránením nepotrebných inštrukcií a optimalizáciou ciest zabezpečujú RISC vynikajúci výkon za zlomok energetickej náročnosti zariadení CISC (Complex Instruction Set Computing). Vďaka redukovanému množstvu inštrukcií vyžadujú ARM procesory menší počet tranzistorov, čo vedie k menším rozmerom. Kombináciou vlastností, ku ktorým patria malé rozmery a nízka spotreba, nachádzajú široké uplatnenie hlavne v mobilných zariadeniach, tabletoch a vstavaných systémoch.

2.4.2 ARM architektúra

V súčasnosti patrí ARM k najviac rozšírenej architektúre procesorov, ktorá obsahuje 44 základných inštrukcií s jednotnými a pevne nastavenými inštrukčnými poľami o veľkosti 32 bitov na zjednodušenie dekódovania. Štrukturálne obsahuje ARM veľký jednotný registračný súbor, do ktorého sa nahrávajú a ukladajú operácie pre spracovanie údajov pracujúcich len v rámci obsahu registra, nie obsahu pamäte. Ku každému údaju načítaného z obsahu registra je priradená adresa založená na jednoduchých adresných režimoch. Okrem iného ARM architektúra zabezpečuje ovládanie ALU (Arithmetic Logic Unit). Pre optimalizáciu programových slučiek zabezpečuje automatické inkrementovanie a dekrementovanie adresy. A nakoniec zaisťuje aj načítavanie a ukladanie viacerých pokynov na maximalizáciu dátovej priepustnosti a podmienené vykonanie takmer všetkých pokynov na maximalizáciu výkonu. Viac o ARM architektúre v [12].

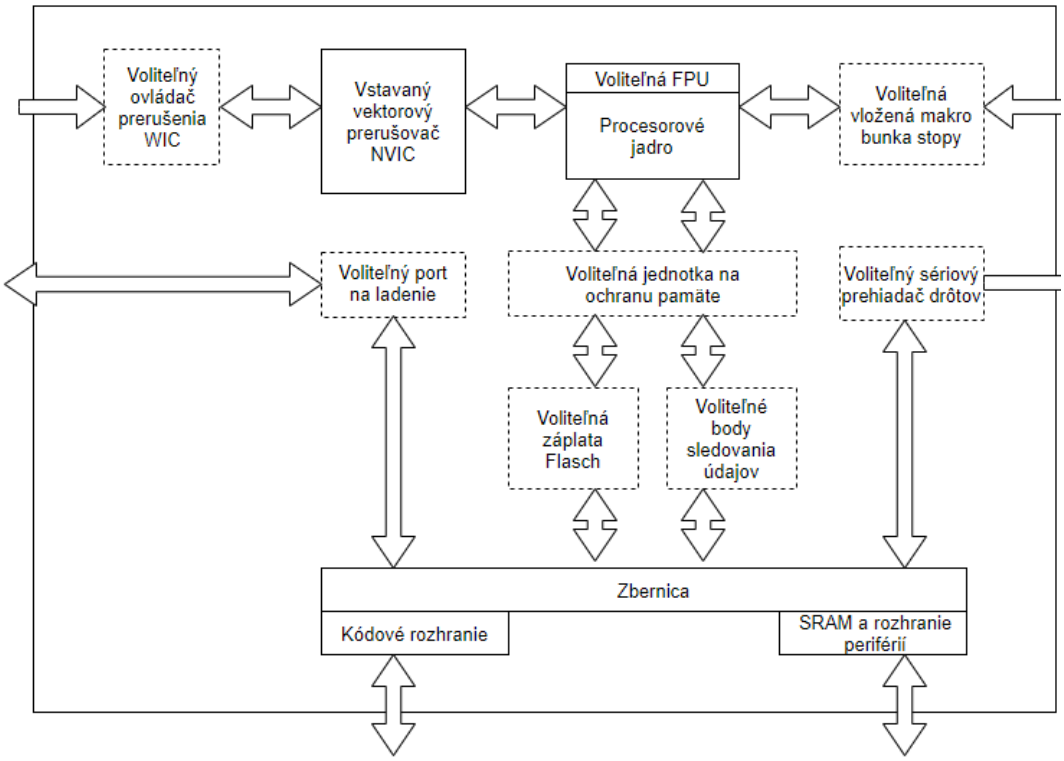
ARM architektúra bola základom k vytvoreniu viacerých jadier procesorov. Medzi najvýznamnejšie patria ARM Cortex, ktoré sa ďalej členia na Cortex-A, Cortex-R a Cortex-M. Cortex-A sa vyznačuje vysokým výkonom procesora a schopnosťou plného využitia operačného systému, preto sa najviac aplikuje vo sfére mobilných telefónov či digitálnych televízií. Cortex-R nachádza uplatnenie v tzv. „real-time“ sfére, kde využíva vysoký výkon a spoľahlivosť. V tomto prípade sa využitie pohybuje napr. v oblasti automobilových brzdných systémov. Posledný typ procesora Cortex-M je nákladovo citlivejší pre deterministické mikrokontroléry. Aplikuje sa v širokej sfére inteligentných meracích systémov, automobilovej elektronike, či v lekárskej oblasti. Bližšie informácie k tomuto odstavcu sa nachádzajú v [13].

2.4.3 Cortex-M4 procesor

Tento typ procesora bol zavedený v roku 2010 a patrí medzi vysoko účinné vbudované procesory vyvinuté tak, aby riešili digitálne riadenie signálov, ktoré vyžadujú efektívnu a ľahko použiteľnú kombináciu možností ovládania a spracovania signálov [13]. Cortex-M4 jadro, ako je vidieť na ilustrácii Obr. 2.4-1, je vybavené jednotkou FPU (Floating Point Unit), ktorá podporuje všetky jednoduché (single-precision) ARM pokyny na spracovanie údajov a dátové typy. Implementuje tiež plný súbor pokynov DSP (Digital Signal Processor), ktoré umožňujú efektívne spracovanie signálov a komplexné vykonávanie algoritmov. Taktiež obsahuje jednotku na ochranu pamäte MPU (Memory Protection Unit), ktorá zlepšuje aplikačnú bezpečnosť. Tieto a nasledujúce poznatky sú získané z literatúry [12] a [13].

Tento procesor je navrhnutý podľa harvardskej architektúry, ktorá sa vyznačuje oddelenou pamäťou pre program (inštrukcie) a dáta (premenné). Jadro procesora obsahuje interné registre obsahujúce šestnásť 32 bitových registrov, ALU, dátové cesty a kontrolnú logiku. Jedna z najdôležitejších periférií je radič prerušenia NVIC

(Nested Vectored Interrupt Controller). Hlavná funkcia NVIC je podpora vnorených podporovaných prerušení. Ide o to, že pokiaľ sa vykonáva prerušenie a vyvolá sa prerušenie s väčšou prioritou, tak sa začne vykonávať prioritnejšie prerušenie a po jeho dokončení sa vráti na predchádzajúce. Ďalšou dôležitou periférioou je radič prerušenia budenia WIC (Wakeup Interrupt Controller), ktorý nachádza uplatnenie v nízko výkonových aplikáciách, kde mikrokontrolér môže vstúpiť do spiaceho módu vypnutím väčšiny komponentov. Prepojenie zbernice je ďalší komponent umožňujúci prenos dát na rôznych zberniciach súčasne. Taktiež poskytuje správnosť prenosu údajov. Poslednou dôležitou periférioou procesora je ladiaci podsystém, ktorý ovláda riadenie ladenia. Keď nastane problém, jadro procesora prejde do pozastaveného stavu, v ktorom vývojári môžu analyzovať stav procesora v danom bode.



Obr. 2.4-1 Blokový diagram jadra Cortex-M4 [13]

2.4.4 Mikrokontrolér STM32

Zariadenia STM32 sú mikrokontroléry s veľmi nízkym výkonom založené na vysoko účinnom 32 bitovom jadre ARM Cortex-M4 s RISC architektúrou. STM32 obsahujú vysokorýchlostné pamäte, širokú škálu vylepšených I/O periférií a rozhranie Quad SPI, ktorý obsahuje ďalšie dve I/O linky, ktorými vysiela súčasne 4 dátové bity počas jedného hodinového cyklu. Pracujú maximálne s frekvenciou $80\,MHz$. Tieto mikrokotroléry ponúkajú veľké množstvo komunikačných sériových a paralelných

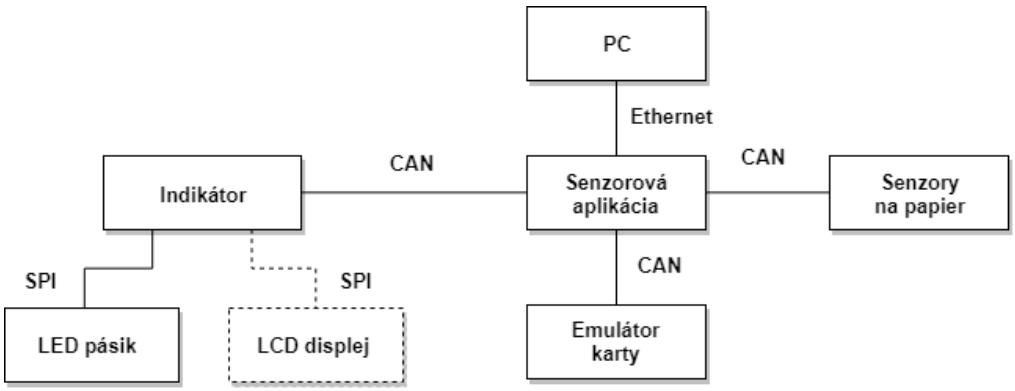
periférií, ktoré môžu byť prepojené s rôznymi druhmi elektronických komponentov. STM32 poskytujú niekoľko ochranných mechanizmov proti zápisu alebo čítaniu kódu z pamäte. Ponúkajú tiež rýchly 12 bitový ADC prevodník, dva komparátory, jeden operačný zosilňovač, dva DAC kanály, RTC (Real Time Clock) s nízkym výkonom, jeden univerzálny 32 bitový časovač, jeden 16 bitový PWM (Pulse Width Modulation) časovač určený na riadenie motora, štyri univerzálne 16 bitové časovače a dva 16 bitové časovače s nízkym výkonom. Bližšie informácie k mikrokontrolérom STM32 sú obsiahnuté v zdrojoch [14] a [15].

2.5 Vstavané systémy

Nasledujúca kapitola o vstavaných systémoch čerpá informácie z literatúry [16] a [17]. Embedded systém je vstavaný systém tvorený kombináciou počítačového hardwaru a softwaru, poprípade je tvorený ďalšími špecializovanými mechanickými či elektronickými časťami. V podstate ide o spoľahlivý systém riadenia v reálnom čase založený na báze mikrokontroléra riadeného softwarom. Takže jeho tri základné komponenty sú hardware, aplikačný software a „real time“ operačný systém, ktorý poskytuje mechanizmus na spustenie procesu. Všeobecne platí, že ukladanie programov a operačných systémov na vstavaných zariadeniach využíva flash alebo prepisovateľnú pamäť typu flash. Súčasťou vstavaných systémov je fimware, čiže špecifický softvér slúžiaci na ovládanie rôznych funkcií zariadenia a systému.
Vstavané systémy vykonávajú zvyčajne jednu funkciu ako napr. bezdrôtový router, ktorého úlohou je posielať dáta. Tieto zariadenia sú úzko obmedzené, takmer všetky diely sa musia nachádzať na jednom čipe, pričom musia vykonávať dostatočne rýchlo operácie v reálnom čase so spotrebovaním minima výkonu na predĺženie životnosti batérie. Ich reakcia na zmeny v systéme musí dosahovať presný čas bez oneskorenia, čo je dôležité napr. v automobilovom priemysle, kde monitorovanie a reakcia na jednotlivé snímače musí byť presná, aby nedošlo ku kolízií. Výhodou týchto zariadení je ich prispôsobivosť, vylepšený výkon, nízka spotreba a nízke náklady.

3. RIEŠENIE

Táto bakalárska práca pozostáva z návrhu a realizácie indikátora stavu robotického systému, ktorého postavenie v systéme spolu s typom komunikácie na jednotlivých periférіach je možné vidieť na obrázku Obr. 2.5-1. Indikátor podobne ako ďalšie komponenty senzorického systému komunikuje so systémovým rozhraním, senzorovou zbernicou, pomocou protokolu CAN a súčasne komunikuje s jednotlivými komponentami pomocou sériového periférneho rozhrania.



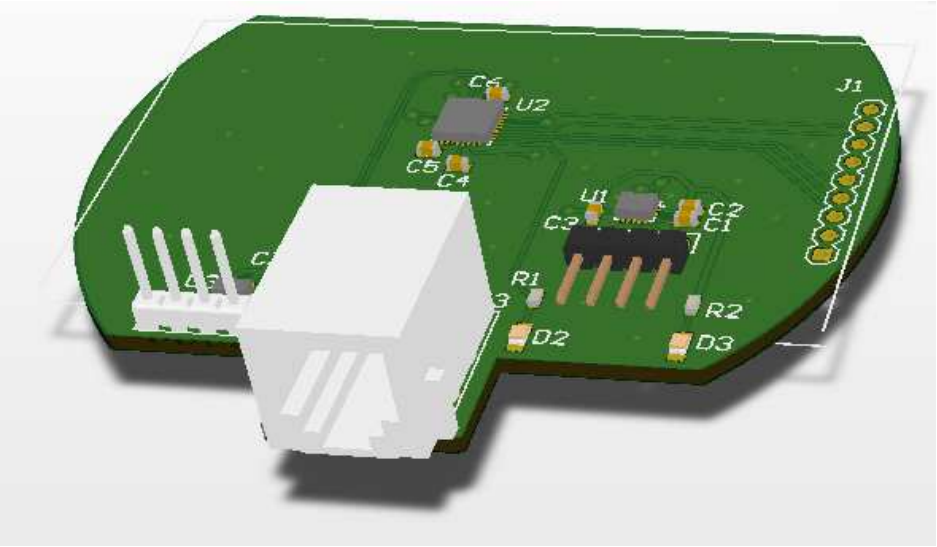
Obr. 2.5-1 Bloková schéma

Riešenie tejto bakalárskej práce pozostáva v podstate z dvoch častí. Prvá časť je tvorená samotným návrhom hardvérovej časti, t. j. návrh dosky plošných spojov, výber vhodných súčiastok, návrh samotného zariadenia s víziou konečnej realizácie, ktorá by bola čo najviac kompatibilná s už existujúcim senzorickým systémom. Je možné sem zaradiť informácie o využitých komponentoch a signalizačnom prvku. Aby táto kostra hardvérových komponentov nadobudla funkčnosť, tak je neoddeliteľnou súčasťou druhá softvérová časť. Jej náplňou je naprogramovanie samotného mikrokontroléra, komunikácie medzi signalizačným prvkom a mastrom (MCU) ako aj nadefinovanie podmienok pre funkčnosť zariadenia po integrácií do systému.

3.1 Hardvérové riešenie

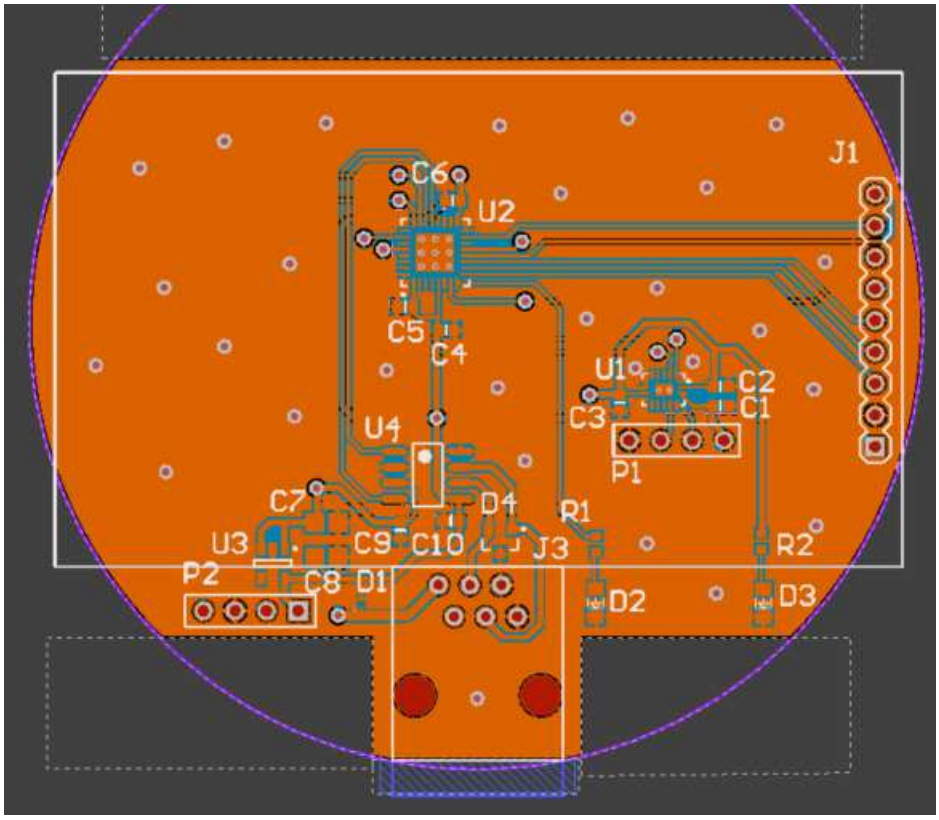
3.1.1 Doska plošných spojov

DPS bola navrhnutá v programe CircuitMaker spoločnosti Altium [18]. Jedná sa o dostupný freeware, ktorý slúži ako schematický a návrhový nástroj. Tento program je navrhnutý tak, aby bola práca s ním čo najviac intuitívna a nenáročná. CircuitMaker tvorí komunita dizajnérov, tvorcov alebo aj študentov, ktorí spolupracujú na vytváraní produktov. Tento program umožňuje zobraziť aj jednoduché 3D modely navrhovaných plošných spojov.



Obr. 3.1-1 3D model DPS pre indikátor navrhnutý v programe CircuitMaker

Táto doska je jednostranne osadená, ale má na oboch stranách vytvorený vodivý obrazec. Vzájomné prepojenie vrstiev je zabezpečené prostredníctvom prekovených otvorov. Priamo na plošnom spoji sú umiestnené konektory, a to konektor pre napájanie, konektor na programovanie pomocou SWD (Serial Wire Debug) protokolu a konektory pre LED reťazec a LCD displej. Samotný tvar dosky je prispôsobený plastovej trubici, ktorá bude tvoriť kryt LED pásika. Na nasledujúcom obrázku Obr. 3.1-2 je možné vidieť navrhnutý plošný spoj v programe CircuitMaker. Je tu uvedený pohľad zhora na ilustráciu rozloženia jednotlivých súčiastok s prídavnou predstavou o uložení LCD displeja.

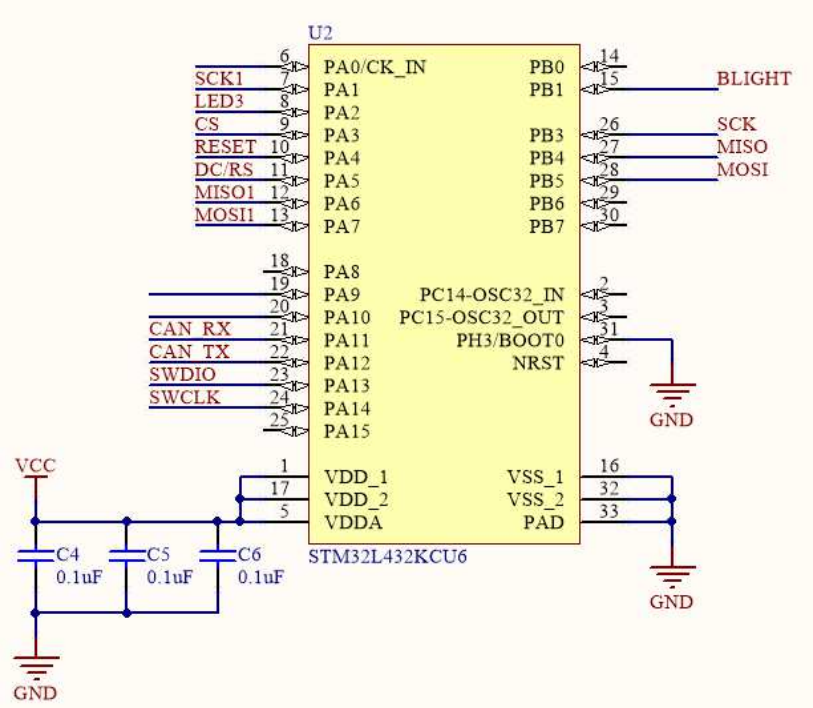


Obr. 3.1-2 Navrhnutá DPS – pohľad zhora

Základom plošného spoja je mikrokotrolér, ktorý slúži na riadenie aplikácie vbudovanej do zariadenia. Jeho hlavnou úlohou je riadenie LED reťazca a v budúcnosti aj LCD displeja. Na základe požiadaviek firmy bol vybraný nízko výkonný mikrokontrolér STM32L432KC s jadrom ARM Cortex-M4. Výber tohto mikrokontroléra je výhodný z oblasti vývoja vstavaných aplikácií, vďaka firme STMicroelectronics, ktorá vytvorila konfiguračný nástroj STM32CubeMX, pomocou ktorého je možné nakonfigurovať mikroprocesory STM32 v grafickom rozhraní a následne vytvoriť zodpovedajúci kód v jazyku C. Ďalším benefitom výberu tohto mikrokontroléra je aj existujúci pomocný vývojový produkt tejto firmy, vývojová doska STM32 Nucleo, ktorá obsahuje nástroj pre odladenie chýb a programátor, čo značne urýchli tvorbu vyvíjaného produktu.

Čo sa týka zapojenia, tak k procesoru sú paralelne zapojené blokovacie kondenzátory s kapacitou $100\ nF$. Procesor je napájaný napätím $3{,}3\ V$, ktoré je pomocou regulátora LP5907MFX pretransformované z $5\ V$. Ide o lineárny regulátor navrhnutý tak, že pracuje so vstupným $1\ \mu F$ a výstupným $1\ \mu F$ keramickým kondenzátorom na oddelenie šumu [19]. V obvode sú zakomponované aj dve LED, ktorých svietivý stav signalizuje pripojené napájanie a funkčnosť mikrokontroléra. Na ochranu pred elektrostatickým výbojom je v obvode použitá ESD (ElectroStatic Discharge) dióda D1. Ďalšou z výhod tohto mikrokontroléra je široká škála programovateľných I/O periférií a taktiež možnosť využitia viacerých sériových a paralelných

komunikačných rozhraní, ktoré sú využité na komunikáciu so signalizačnými prvkami. Ako je vidieť na obrázku Obr. 3.1-3, využívané sú periférie slúžiace na sériovú komunikáciu, komunikáciu cez rozhranie CAN a piny GPIO (General Purpose Input/Output) pre ovládanie vlastností LCD displeja. Na preprogramovanie, prípadne vylaďovanie chýb firmwara procesora, slúžia signály $SWDIO$ a $SWCLK$. Tieto signály sú súčasťou SWD protokolu, čo je dvojpinové elektrické rozhranie a sú prepojené s pomocnou vývojovou doskou Nucleo, prostredníctvom ktorej je možné preprogramovať MCU.



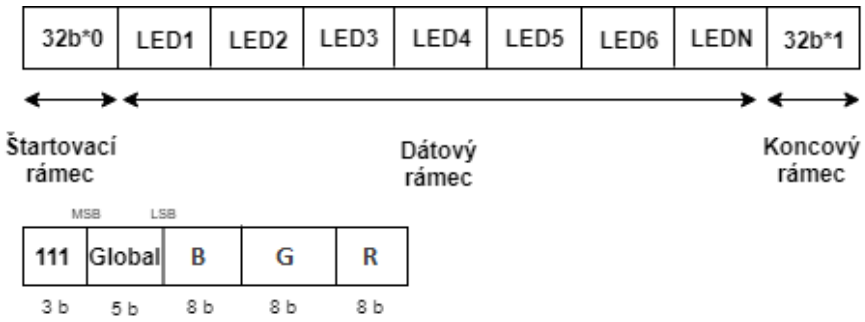
Obr. 3.1-3 Schéma procesora tvoriaceho indikátor v programe CircuitMaker

Keďže mikroprocesor má zabudovaný len CAN kontrolér, je potrebné pre správnu komunikáciu s CAN zbernicou vyžadujúcou diferenciálnu úroveň signálov, pripojiť aj CAN vysielač/prijímač. Na jeho ochranu vo vysokorýchlostných sieťach a ESD výboja je zakomponovaná ochranná dióda D4. Piny CAN_RX a CAN_TX sú pripojené na CAN vysielač/prijímač IFX1050G VIO [20], ktorý vysiela dáta na zbernicu. Využíva napájanie v rozmedzí od $3{,}3\ V$ do $5\ V$. Celková obvodová schéma sa nachádza v prílohe1.

3.1.2 Signalizačné periférie

Na signalizáciu stavu bol vybraný LED trojfarebný RGB stmievajúci pásik APA102 [21]. Svoje uplatnenie nachádza v oblasti, kde je potrebné zdôrazniť osvetlenie, takže je vhodným nástrojom pre signalizáciu. Tiež je tvorený integrovaným obvodom vytvoreným procesom CMOS (Complementary Metal Oxide Semiconductor)

poskytujúcim ovládanie trojfarebného RGB LED výstupu, ktorým je možné nastavovať sýtosť výstupu na škále s 256 úrovňami (8 bitov) a nastavovať jas na 32 úrovňovej škále (5 bitov). Keďže je potrebné, aby bol pásik napájaný $5\,V$ a mikrokontrolér pracuje s napätím $3{,}3\,V$, tak je v obvode použitý štvorbitový prevodník napätia NTB0105BQ-Q100X firmy Semiconductor [23].
Použitý LED pásik komunikuje s MCU prostredníctvom dvojvodičového SPI rozhrania. Konkrétne využíva riadiaci hodinový signál SCK a dátový signál $MOSI$. Princíp vysielania dát (Obr. 3.1-4) v tomto pásiku pozostáva v rozdelení dátových polí do troch rámcov: štartovací, dátový a koncový rámec.



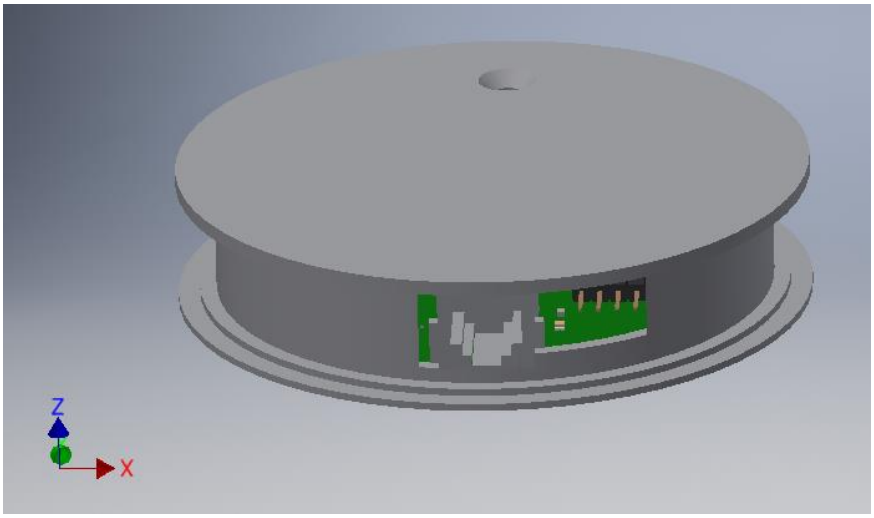
Obr. 3.1-4 Diagram APA102 protokolu [22]

Začiatočný rámec tvorí 32 bitov, ktoré sú nastavené do logickej nuly. Nasledujúci dátový rámec sa skladá z piatich polí. Prvé trojbitové pole je vždy nastavené na samé jednotky. Po ňom nasleduje päťbitové pole, ktorým sa nastavuje jas LED pásika, pričom maximálny jas je nastavený pri hodnote 32, t. j. každý bit je nastavený do logickej jednotky. Posledné tri polia slúžia na nastavenie výslednej farby a sýtosti. Každé pole je tvorené ôsmimi bitmi, takže nastavovací rozsah sýtosti je od 0 do 255 pre každú farbu. Úlohou posledného rámca je dodať do reťazca viac hodinových impulzov, až kým údaje nepreniknú do poslednej LED v reťazci, pričom požadovaný počet impulzov je aspoň polovica celkového počtu LED v reťazci. Takže celkové nastavenie LED pásika pozostáva z nastavenia začiatočného 32 bitového rámca na nuly, 32 bitového dátového rámca určujúceho farbu pre každú LED a koncového rámca nastaveného do logickej jednotky [22].

3.1.3 Design zariadenia

Pri návrhu indikátora stavu je potrebné brať najmä ohľad na viditeľnosť daného zariadenia a jasné efekty, ktoré musia byť zreteľné. Samotná DPS má netypický tvar, ktorý bol prispôsobený tvaru plastovej trubky s kruhovým prierezom. Táto trubka je súčasťou držiaka DPS vytvoreného 3D tlačiarňou. Na návrh bol použitý parametrický 3D modelový softvérový CAD (Computer Aided Design) Inventor firmy Autodesk.

V Inventori boli navrhnuté dve časti krytky. Spodná časť, do ktorej sa vloží DPS a vrchná, ktorá slúži na zakrytie DPS. V krytke je vyrezaný jeden otvor slúžiaci na vyvedenie kábla pre pripojenie do senzorovej zbernice a pre vyvedenie LED pásika na obvod krytky vytvorenej v 3D tlačiarni. Táto vymodelovaná časť slúžiaca pre uchytenie LED pásika je tvorená vrypom po obvode, do ktorého sa zachytí trubica z matného plexiskla, ktorá zabezpečí jednak ochranu svetelného reťazca, ale aj lepšiu vizualizáciu finálneho osvetlenia. Na nasledujúcom obrázku (Obr. 3.1-5) je vizualizácia obalu DPS navrhnutá v aplikácii Inventor. Po pridaní LED pásika ide v podstate o finálne 3D zobrazenie indikátora stavu robotického systému.



Obr. 3.1-5 Model krytky navrhnutý v aplikácii Inventor

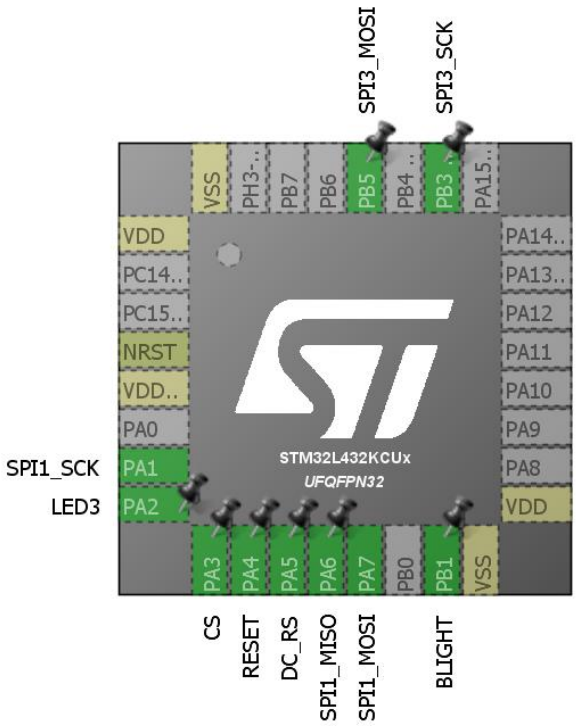
3.2 Softvérové riešenie

Na implementáciu riešenia je použitý imperatívny programovací jazyk C, ktorým je naprogramovaný firmware. V podstate ide o program, ktorý interne zabezpečuje funkčnosť elektronického zariadenia. Tento firmware je doplnený softvérom využívajúcim vysokoúrovňový objektovo orientovaný programovací jazyk C#. Kompletná softvérová realizácia je uskutočnená vo vývojovom prostredí Visual Studio. Naprogramované zdrojové súbory sú súčasťou CD prílohy.Príloha 4 - Zdrojové kódy

3.2.1 STM32CubeMX

Ako už bolo spomenuté, firma STMicroelectronics vytvorila konfiguračný nástroj STM32CubeMX [24], pomocou ktorého je možné jednoducho a rýchlo nakonfigurovať mikroprocesory STM32. Tento nástroj poskytuje grafické užívateľské rozhranie (Obr. 3.2-1) užitočné hlavne na nastavenie jednotlivých registrov. Ponúka tiež možnosť nastavenia zdroja hodinového signálu pre jednotlivé

zbernice alebo nastavenie komunikačných periférii ako sú SPI, I2C alebo CAN. Veľkou výhodou tohto nástroja je generácia inicializačného kódu v programovacom jazyku C. Využíva vrstvu HAL (Hardware Abstraction Layer), ktorá slúži ako rozhranie umožňujúce pridávať podporu pre nové zariadenia a nové spôsoby pripojenia zariadení k počítaču bez toho, aby sa zmenila každá aplikácia, ktorá zariadenie používa. Ide o vysokoúrovňové a funkčne orientované API (Application Programming Interface).



Obr. 3.2-1 Konfigurácia MCU indikátora v STM32CubeMX

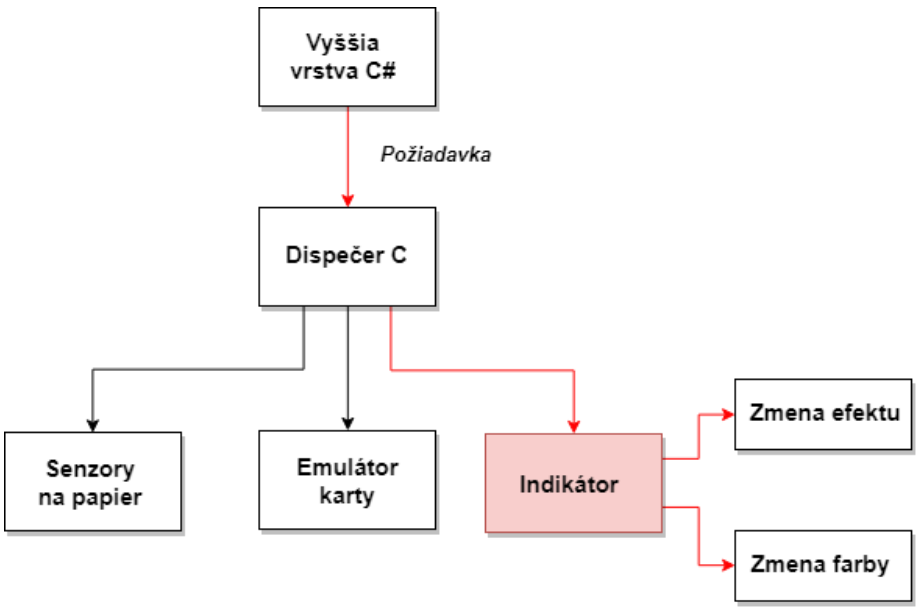
3.2.2 Implementácia firmwaru

Prvým krokom pre vytvorenie firmwaru vyvíjaného zariadenia bolo vygenerovanie inicializačného kódu z aplikácie STM32CubeMX podľa nastavenej konfigurácie. Tento kód tvorí základnú kostru pre následné naprogramovanie periférii MCU. Využitím existujúcej konfigurácie sériového periférneho rozhrania bola vytvorená inicializácia komunikácie medzi MCU a LED pásikom.

Následne bol vytvorený kód slúžiaci na ovládanie a nastavovanie LED reťazca pomocou APA102 protokolu (Obr. 3.1-4), ktorý sa skladá z nastavenia jednotlivých rámcov značiacich začiatok, koniec dátového toku a samotné dáta obsahujúce informáciu o farbe a jase, ktoré majú byť nastavené. Po vytvorení LED protokolu bolo možné implementovať jednotlivé funkcie definujúce rôzne stavy indikátora. Dané signalizačné zariadenie oznamuje o zmene stavu robotického systému pomocou zmeny farby a tiež boli vytvorené doplňujúce funkcie určujúce signalizačné efekty. Prvým efektom je kontinuálne svietenie celého LED reťazca. Rozdielnosť stavov je

daná zmenou farby nastavenou podľa potreby užívateľa. Ďalším efektom je konštantné blikanie celého reťazca a posledný efekt znázorňuje postupné rozsvietenie jednej LED v reťazci, pričom ostatné sú zhasnuté, v podstate ide o vytvorenie „hadieho“ efektu. Toto sú varianty, ktoré je možné použiť pre LED pásik na signalizáciu stavov, pričom je možné doplniť tieto efekty o iné v závislosti od potreby užívateľa.

Po zabezpečení komunikácie s MCU a zabezpečení funkčnosti APA102 protokolu je dôležitou úlohou pri zakomponovaní nového zariadenia do senzorického systému samotná definícia vlastností zariadenia. Aby boli jednotlivé periférie všetkých prídavných zariadení odlíšené, využíva sa už existujúci modulárny systém, v ktorom sa nadefinuje typ novo pridaného zariadenia a spôsob, akým sú tieto koncové body obsluhované. Z tohto dôvodu je nadefinovaný tzv. dispečer (Obr. 3.2-2), ktorého úlohou je obsluha konkrétneho adresovaného zariadenia. Dispečer je tvorený štruktúrou obsahujúcou ukazovatele na funkcie, ktoré sú zavolané v prípade, ak dôjde k nejakej udalosti. Konkrétne pri štarte zariadenia sa zavolá funkcia, ktorá inicializuje indikátor. Je tu aj funkcia na čítanie, ktorá naplní definovanú štruktúru a následne vráti dáta užívateľovi. V tomto type zariadenia nie je zatiaľ nutné využitie tejto funkcie. Ďalšia funkcia, obsahujúca pole pre zápis, naplní štruktúru prijatými informáciami o stave prijatých dát a operácií, ktoré majú byť prevedené a následne zavolá konkrétnu akciu. Ide napr. o situáciu, kedy príde požiadavka na zmenu farby na indikátore. Posledná funkcia tejto štruktúry má za úlohu zaobstarať pamäťové pole pre zapísanie dát, poprípade vyčítanie dát z tohto poľa. Pri príjme požiadavky firmware rozpozná dispečera a pomocou popisu pamäťového poľa zistí, či dané pole existuje a či je na ňom možné vykonať požadovanú akciu. V závislosti od požiadavky z vyššej vrstvy dôjde k nastaveniu konkrétneho efektu pre definovaný stav.

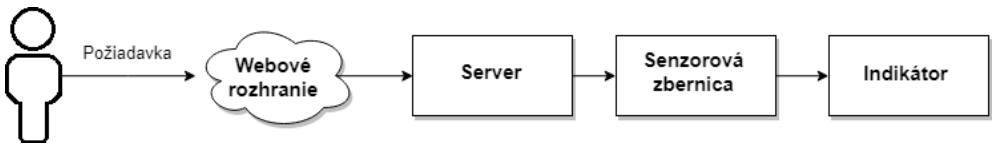


Obr. 3.2-2 Diagram znázorňujúci funkciu dispečera

3.2.3 Implementácia softvéru

Dôležitú úlohu pri komunikácii s prvkami senzorického systému zohráva server, ktorý spracuje dáta prijaté od užívateľa a odošle ich do firmwaru mikroprocesora. Je potrebné pripomenúť, že aplikácia servera je vyvinutá v programovacom jazyku C# s využitím frameworku .NET Core.

Je dobré si najskôr ujasniť funkciu servera, ktorý nachádza využitie pri propagácií požiadaviek od klienta k požadovanej periférii, v tomto prípade k indikátoru stavu. Funkciou servera je tiež zabezpečiť správnu komunikáciu s prevodníkom CAN na USB, ktorý sa využíva v tomto systéme. Tiež rieši implementáciu komunikačného protokolu založeného na báze CAN protokolu a zabezpečuje existenciu prístupových bodov cez REST API (Representional State Transfer Application Programming Interface), ktorá je súčasťou ASP.NET Coru. REST API [25] je softvérová podpora pre vývoj webových aplikácií. Vďaka tejto platforme je možné jednoducho vytvoriť API metódu pre indikátor. V podstate ide o spôsob komunikácie so softvérom. Prostredníctvom tejto časti je možné vytvoriť prístup k serverovej strane, v ktorej nastáva rozšírenie požiadaviek od klienta na konkrétny typ senzora. Výhodou tejto vyššej vrstvy v programovacom jazyku C# je pridanie ďalších funkcionalít k indikátoru tak, že nie je nutné ju nahrávať do MCU, ale stačí ju len nahrať do serverovej časti. Vo výsledku je samotné aktualizovanie jednoduchšie, pretože na jednom serveri môže byť viacero indikátorov, ktoré nebude potrebné aktualizovať osobitne. Na obrázku Obr. 3.2-3 je zobrazená komunikačná schéma medzi jednotlivými komponentami sprostredkúvajúcimi prenos dát od užívateľa po navrhované zariadenie.

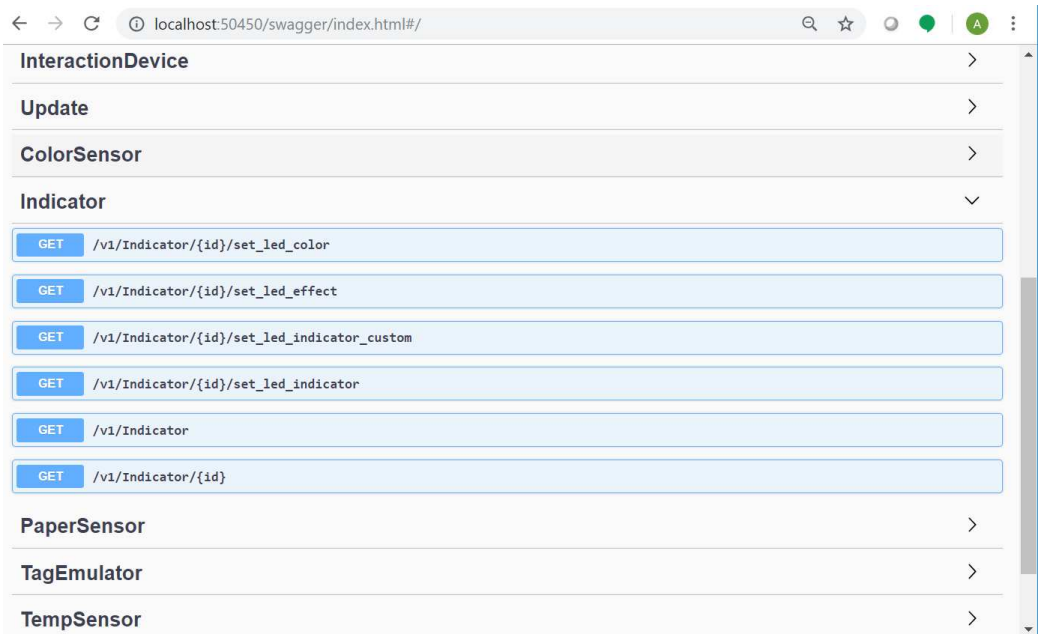


Obr. 3.2-3 Schéma komunikácie medzi jednotlivými komponentami

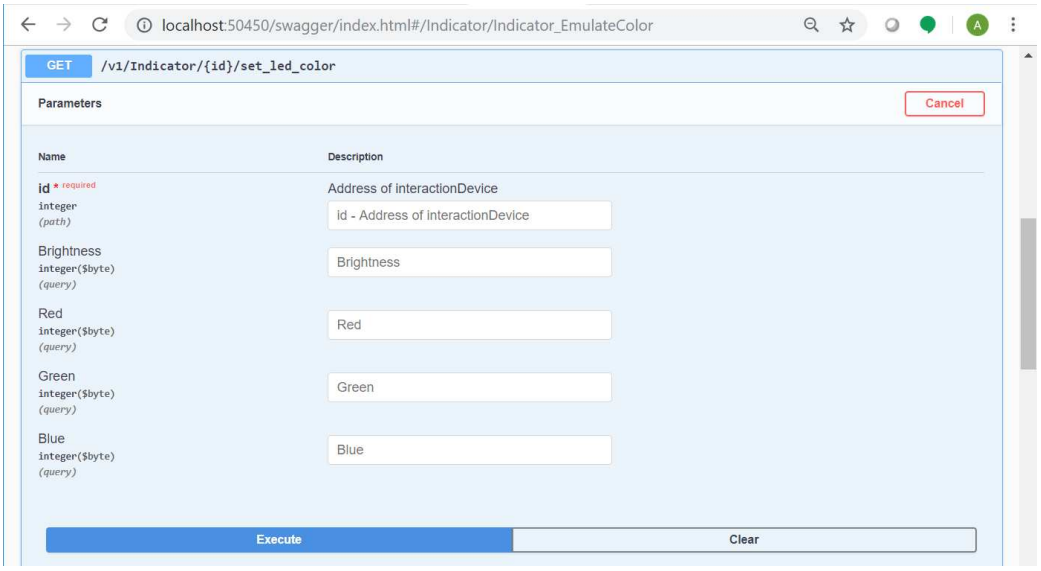
Táto platforma obsahuje štruktúry jednotlivých stavov, do ktorých sa môže robotický systém dostať. Obsahuje rovnako štruktúru efektov, ktorými môžu byť stavy zobrazené a štruktúru farieb, ktoré je možné nastaviť na LED pásiku. Základom je trieda, ktorá zabezpečí nastavenie vlastností pre koncový bod indikátora. Táto časť priamo komunikuje s firmwarom, takže slúži ako prepojovací bod medzi vyššou a nižšou vrstvou softvéru zariadenia.

Keďže C# vrstva zabezpečuje ľahší prístup užívateľa k nastaveniu, bolo vytvorené webové rozhranie pomocou knižnice, ktorá vygeneruje dokumentáciu pre REST API aplikáciu. Výber a nastavenie periférie zariadenia je tvorené cestou, ktorá obsahuje identifikačné číslo periférie a operáciu, ktorá má nastať. Takže pri hľadaní chýb

vo funkčnosti zariadenia alebo pri ďalšom vývoji indikátora je možné využiť práve toto rozhranie a skúšať nastaviť farby a efekty na ňom. Samotná implementácia je tvorená triedou zabezpečujúcou pripojenie k danému koncovému bodu a operáciami, ktoré slúžia na bližšie nastavenie správania zariadenia. Konkrétne ukážky webovej stránky s vlastnosťami, ktoré zahŕňajú sú zobrazené na nasledujúcich obrázkoch. Na prvom obrázku Obr. 3.2-4 je zobrazené menu zariadení, ktoré je možné ovládať pomocou webového rozhrania s tým, že je tu zobrazený rozšírený výber možných funkcionalít indikátora. Celkovo je možné nastavovať osobitne farbu na LED pásiku a efekty. Ďalej je možné nastaviť tieto vlastnosti naraz z hľadiska užívateľa a v poslednom prípade je možné nastaviť konkretný stav na indikátore, t. j. aktívny, varovný alebo chybný. Na druhom obrázku Obr. 3.2-5 je zase možné vidieť nastavovaciu lištu pre farby na zariadení.



Obr. 3.2-4 Výber nastavovacích funkcií cez REST API



Obr. 3.2-5 Menu pre nastavenie farby cez REST API

4. ZÁVER

Cieľom tejto bakalárskej práce bolo navrhnúť a skonštruovať zariadenie pre indikáciu stavu robotického systému komunikujúceho pomocou rozhrania CAN a využívajúceho protokol firmy Y Soft. V samotnom návrhu tohto zariadenia sa bral ohľad hlavne na výslednú kompatibilitu so senzorickým systémom.

V úvodnej časti bola uskutočnená rešerš jednotlivých komunikačných prostriedkov. S každým spomenutým typom komunikácie došlo ku kontaktu či už pri návrhu prototypu zariadenia, ktorý bol súčasťou semestrálnej práce alebo pri samotnom finálnom zariadení. Zbernica CAN tvorí základ pre Y Soft protokol pri komunikácií s nadradeným systémom. SPI zbernica slúži pre komunikáciu so signalizačným prvkom a I2C zbernica našla využitie pri návrhu prototypu, ktorého súčasťou bol komponent Arduina kitu, a to LCD displej s I2C adaptérom. Súčasťou rešerše boli aj základné informácie o vstavaných systémoch a mikrokontroléri typu STM32, ktorý tvorí jadro navrhovaného zariadenia.

Samotné riešenie pozostáva z dvoch častí. V prvej kapitole je bližšie špecifikovaný postup návrhu hardvérovej časti zariadenia. Postupne je táto kapitola tvorená chronologickým vývojom riešenia, od návrhu samotnej DPS v programe CircuitMaker, cez popis súčiastok, ktoré zabezpečujú celkovú funkčnosť zariadenia, po samotné signalizačné prvky. Vytvorený návrh DPS je pripravený na zakomponovanie LCD displeja, ktorý by zobrazoval správu o stave. Táto periféria je nechaná na budúci vývoj pre časovú náročnosť a komplikovanosť naprogramovania displeja. Súčasným signalizačným prvkom je LED pásik. Súčasťou návrhu DPS bol aj dizajn tohto zariadenia. Tvar plošného spoja bol prispôsobený tvaru trubice, ktorá bude slúžiť ako kryt pre LED pásik. Na lepší vizuálny efekt bol v programe Inventor navrhnutý obal pre DPS s uchytením LED pásika.

V druhej kapitole bol navrhnutý a zrealizovaný spôsob komunikácie s LED pásikom a nadradeným robotickým systémom pre testovanie zariadení, a to pomocou webových požiadaviek umožňujúcich ľubovoľnú zmenu farieb a efektov LED pásika. Konkrétne bolo vysvetlené softvérové riešenie tohto zariadenia. Táto časť je tvorená chronologickým sledom návrhu. Ako prvé bol využitý konfiguračný nástroj STM32CubeMX, v ktorom boli nastavené jednotlivé registre mikrokontroléra. Následne bol vytvorený v programovacom jazyku C firmware zariadenia, ktorý je spätý s už existujúcim systémom periférií, avšak bolo nutné pretransformovanie a pridanie konkrétnych funkcií pre tento typ periférie. Po naprogramovaní firmwaru indikátora bola implementovaná serverová časť napísaná v jazyku C#. Táto časť tvorí prístupový bod pre ovládanie indikátora skrz REST API.

Výsledné zariadenie bolo integrované do robotického systému, kde bolo otestované v produkčnom behu, a tým bola zároveň otestovaná aj jeho funkčnosť.

Literatúra

[1] BASICS OF THE SPI COMMUNICATION PROTOCOL. Circuit Basics [online]. [cit. 2018-12-02]. Dostupné z: http://www.circuitbasics.com/basics-of-the-spi-communication-protocol/

[2] Back to Basics: SPI (Serial Peripheral Interface). ALL ABOUT CIRCUITS [online]. [cit. 2018-12-02]. Dostupné z: https://www.allaboutcircuits.com/technical-articles/spi-serial-peripheral-interface/

[3] FRÝZA, Tomáš. Řízení sériové komunikace: Mikroprocesorová technika a embedded systémy [online]. UREL, VUT v Brne, 2018 [cit. 2018-11-22].

[4] Bus arbitration in CAN (Controller Area Network). Linkedin [online]. 2017 [cit. 2018-10-10]. Dostupné z: https://www.linkedin.com/pulse/bus-arbitration-can-controller-area-network-abdul-rehman-anwer

[5] Controller Area Network (CAN) Overview. NATIONAL INSTRUMENTS [online]. 2014 [cit. 2018-12-03]. Dostupné z: http://www.ni.com/white-paper/2732/en/

[6] CORRIGAN, Steve. Introduction to the Controller Area Network (CAN) [online]. Texas Instrument, 2016 [cit. 2018-10-10]. Dostupné z: http://www.ti.com/lit/an/sloa101b/sloa101b.pdf

[7] GRIFFIT, John. What do CAN bus signals look like. Texas Istruments [online]. 4.6.2015 [cit. 2019-05-02]. Dostupné z: https://e2e.ti.com/blogs_/b/industrial_strength/archive/2015/06/04/what-do-can-bus-signals-look-like

[8] DUŠEK, Oto. Platforma pro automatické testování vestavěných zařízení. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Doc. Ing. Zdeněk Vašíček, Ph.D.

[9] Sběrnice CAN [online]. VUT FEKT Brno, 2003 [cit. 2018-11-16]. Dostupné z: http://www.elektrorevue.cz/clanky/03021/index.html

[10] I2C-bus specification and user manual. Nxp [online]. [cit. 2018-12-03]. Dostupné z: https://www.nxp.com/docs/en/user-guide/UM10204.pdf

[11] ARM processor. WhatIs [online]. 2015 [cit. 2018-11-16]. Dostupné z: https://whatis.techtarget.com/definition/ARM-processor

[12] ARM Architecture Reference Manual. NYU COMPUTER SIENCE [online]. 2005 [cit. 2018-12-03]. Dostupné z: https://cs.nyu.edu/courses/spring18/CSCI-GA.2130-001/ARM/arm_arm.pdf

[13] The Cortex-M Series: Hardware and Software [online]. [cit. 2018-11-16]. ISBN 1-55937-354-7. Dostupné z: http://www.eas.uccs.edu/~mwickert/ece5655/lecture_notes/ARM/ece5655_chap2.pdf

[14] STM32L432KB STM32L432KC: Datasheet [online]. 2018 [cit. 2018-11-16]. ISBN 1-55937-354-7. Dostupné z: https://www.st.com/resource/en/datasheet/stm32l432kc.pdf

[15] STM32L432KC. Life.augmented [online]. 2018 [cit. 2018-11-16]. Dostupné z: https://www.st.com/en/microcontrollers/stm32l432kc.html

[16] Embedded system. IoT Agenda [online]. 2018 [cit. 2018-11-16]. Dostupné z: https://internetofthingsagenda.techtarget.com/definition/embedded-system

[17] Embedded Systems - Overview. Tutorialspoint [online]. 2018 [cit. 2018-11-16]. Dostupné z: https://www.tutorialspoint.com/embedded_systems/es_overview.htm

[18] About CircuitMaker. CIRCUITMAKER [online]. [cit. 2019-04-23]. Dostupné z: https://circuitmaker.com/About

[19] LP5907 Ultralow-Noise, 250-mA Linear Regulator for RF and Analog Circuits - Requires No Bypass Capacitor. Texas Instrument [online]. [cit. 2019-04-25]. Dostupné z: http://pdf1.alldatasheet.net/datasheet-pdf/view/804620/TI1/LP5907MFX-1.8/NOPB.html

[20] IFX1050GVIO: High Speed CAN-Transceiver. Infineon [online]. 2011-04-08 [cit. 2019-03-04]. Dostupné z: https://www.infineon.com/dgdl/IFX1050GVIO_DS_10.pdf?fileId=db3a304330f68606013141d8eac7569f

[21] APA102C [online]. [cit. 2019-03-09]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/APA102.pdf

[22] Understanding the APA102 “Superled”. Tim's Blog [online]. [cit. 2019-05-02]. Dostupné z: https://cpldcpu.wordpress.com/2014/11/30/understanding-the-apa102-superled/

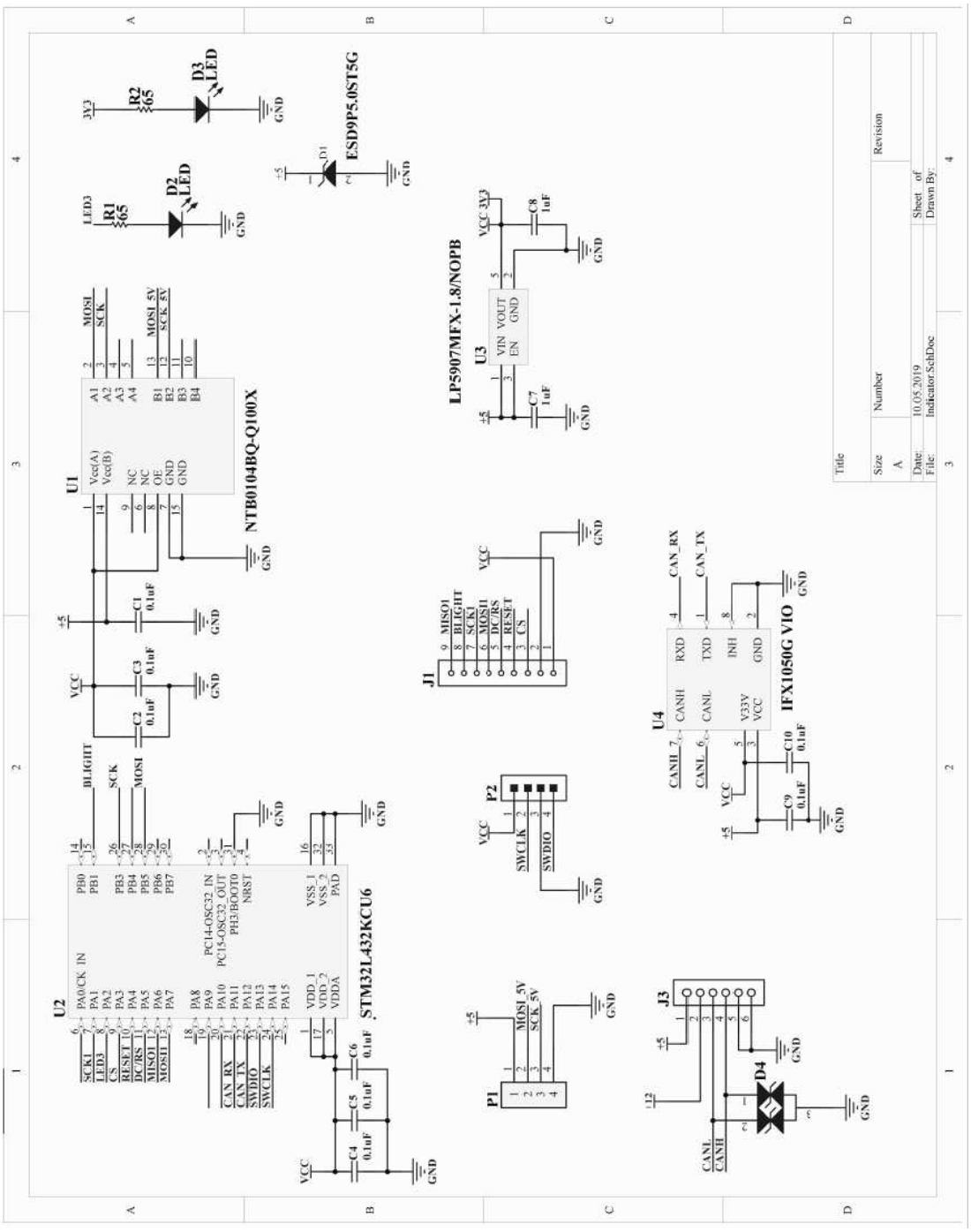
[23] NTB0104-Q100. Nxp [online]. [cit. 2019-03-09]. Dostupné z: https://www.nxp.com/docs/en/data-sheet/NTB0104_Q100.pdf

[24] STM32Cube MCU & MPU Packages. St [online]. [cit. 2019-04-23]. Dostupné z: https://www.st.com/en/embedded-software/stm32cube-mcu-mpu-packages.html

[25] Representational state transfer. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-05-02]. Dostupné z: https://en.wikipedia.org/wiki/Representational_state_transfer

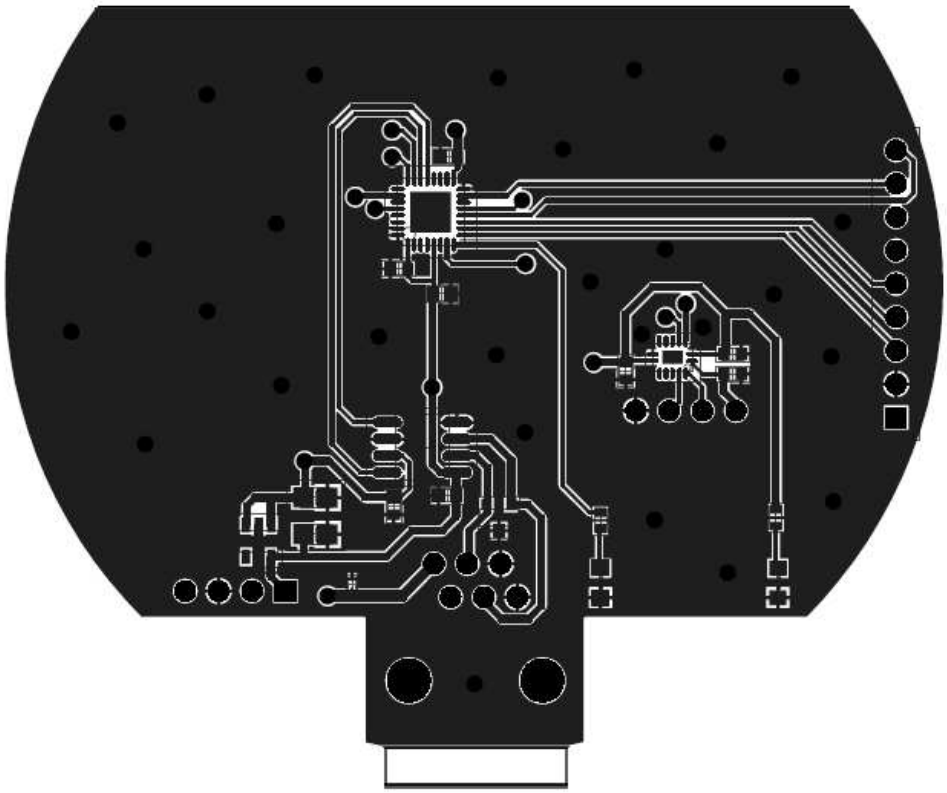
Zoznam príloh

Príloha 1 - Obvodová štruktúra

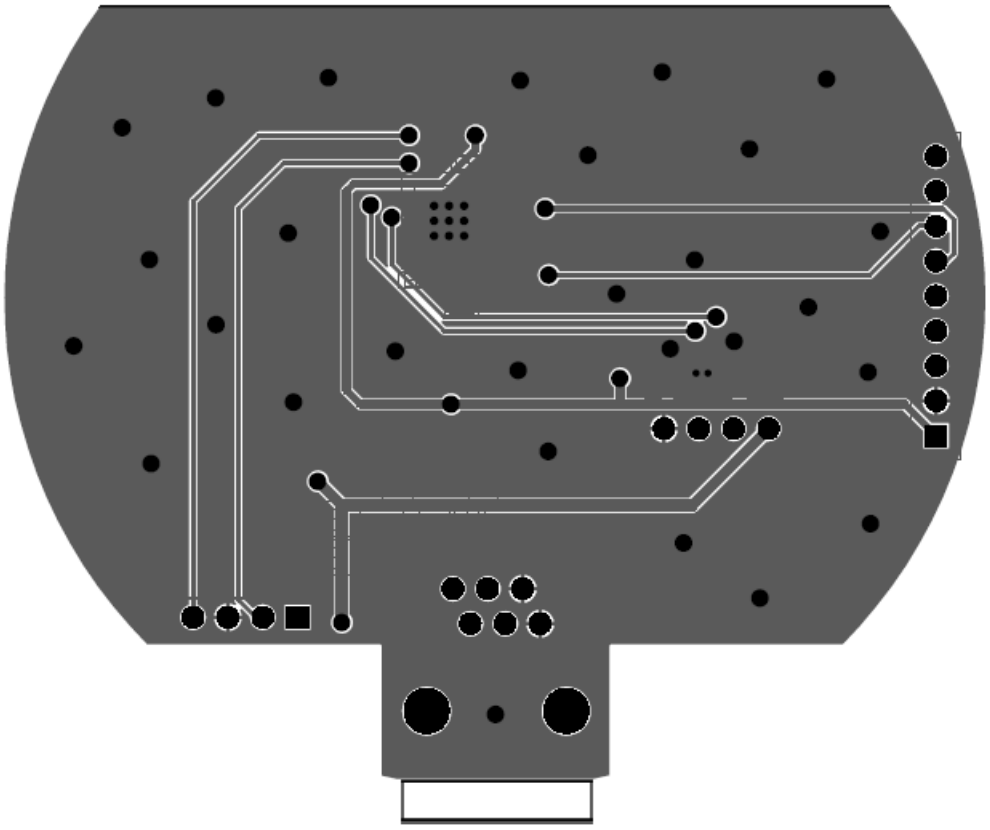


Obr. 3.2-1 Schéma zapojenia

Príloha 2 - Doska plošných spojov



Obr. 3.2-2 DPS – pohľad zhora



Obr. 3.2-3 DPS – pohľad zdola

Príloha 3 - Indikátor

Obr. 3.2-4 Prevedenie indikátora stavu robotického systému

Príloha 4 - Zdrojové kódy

Zdrojové kódy sú súčasťou CD prílohy